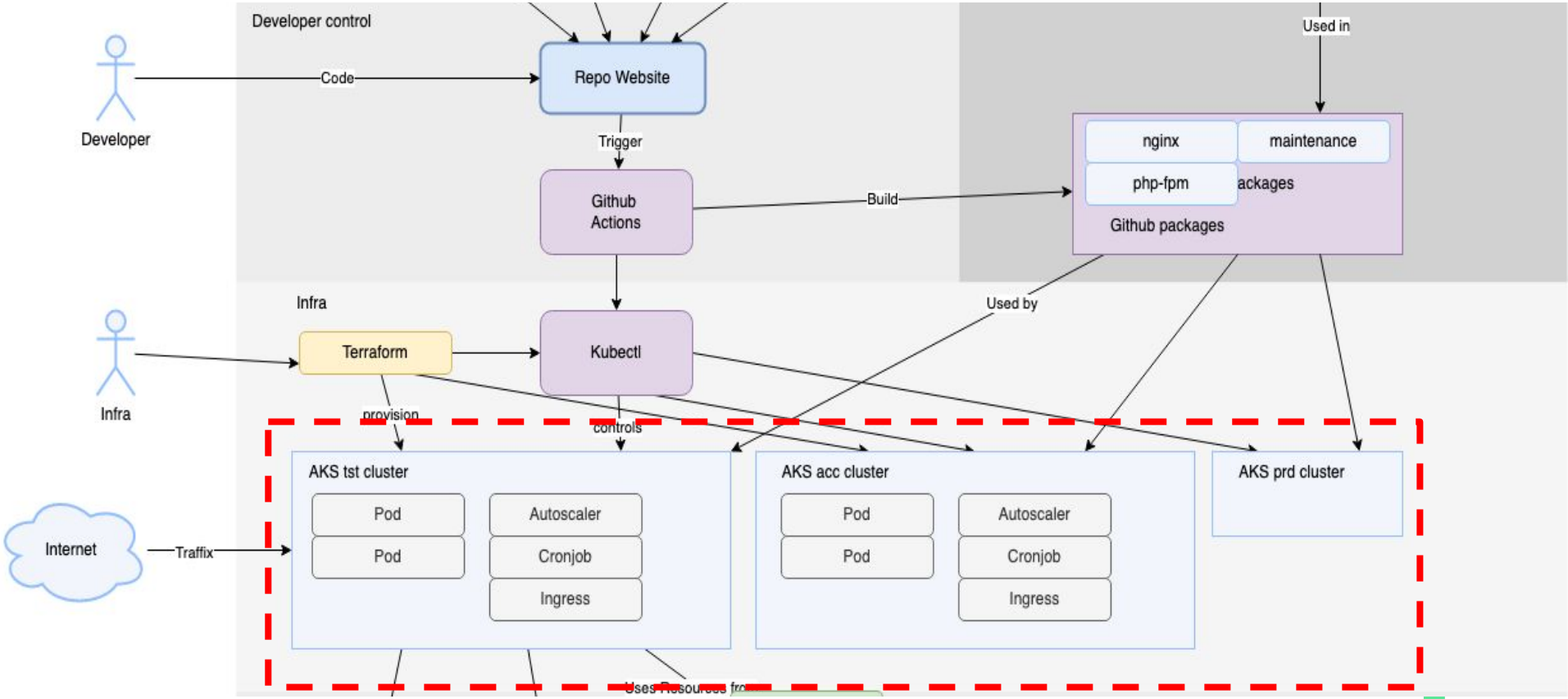
Kubernetes basics

Kubernetes basics

1. Cluster
2. Namespaces
3. Working with a cluster
4. Everything runs on Nodes
5. Pod
6. Service / Deployment
7. Horizontal Pod Autoscaler
8. Deployment strategy
9. Ingress
10. Persistent Volume Claim
11. Configmap
12. Statefullset
13. Job
14. Cron
15. Loadtesting
16. Monitoring
17. Logs

Blogpost
with some
this conten

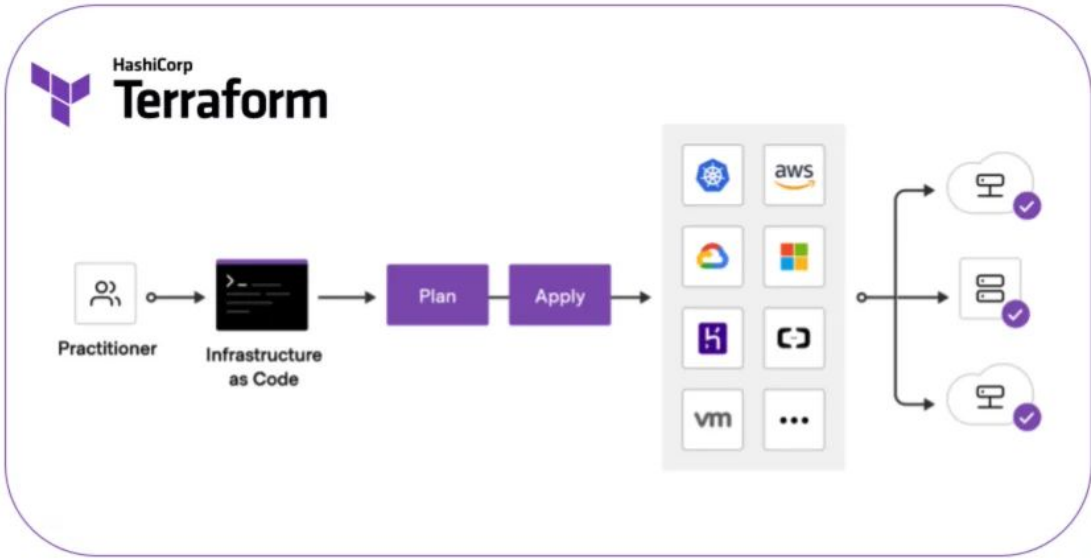
Clusters


Developer control
Developer
Code
Repo Website
Trigger
Github Actions
Build
Used in
nginx
maintenance
php-fpm
ackages
Github packages
Used by
Infra
Infra
Terraform
Kubectl
provision
controls
AKS tst cluster
Pod
Pod
Autoscaler
Cronjob
Ingress
AKS acc cluster
Pod
Pod
Autoscaler
Cronjob
Ingress
AKS prd cluster
Internet
Traffix

Cluster creation

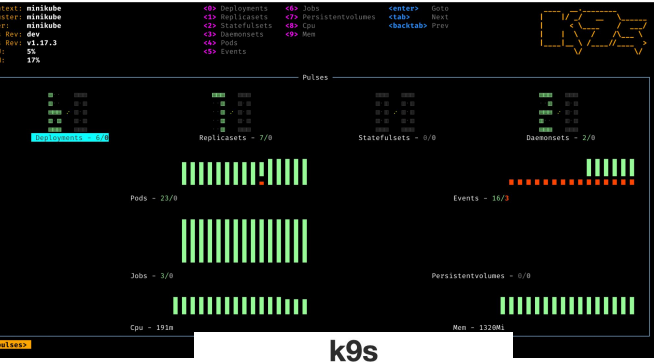
Manual

in Code



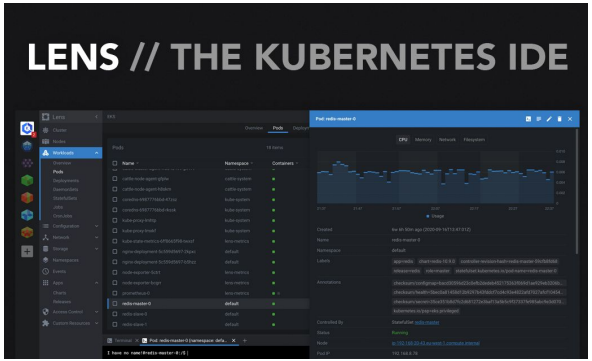
Working with clusters

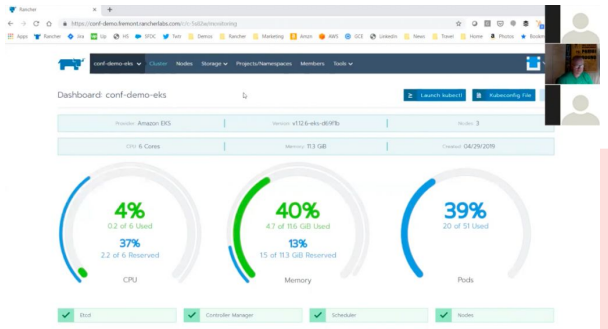
Tools for the job



k9s

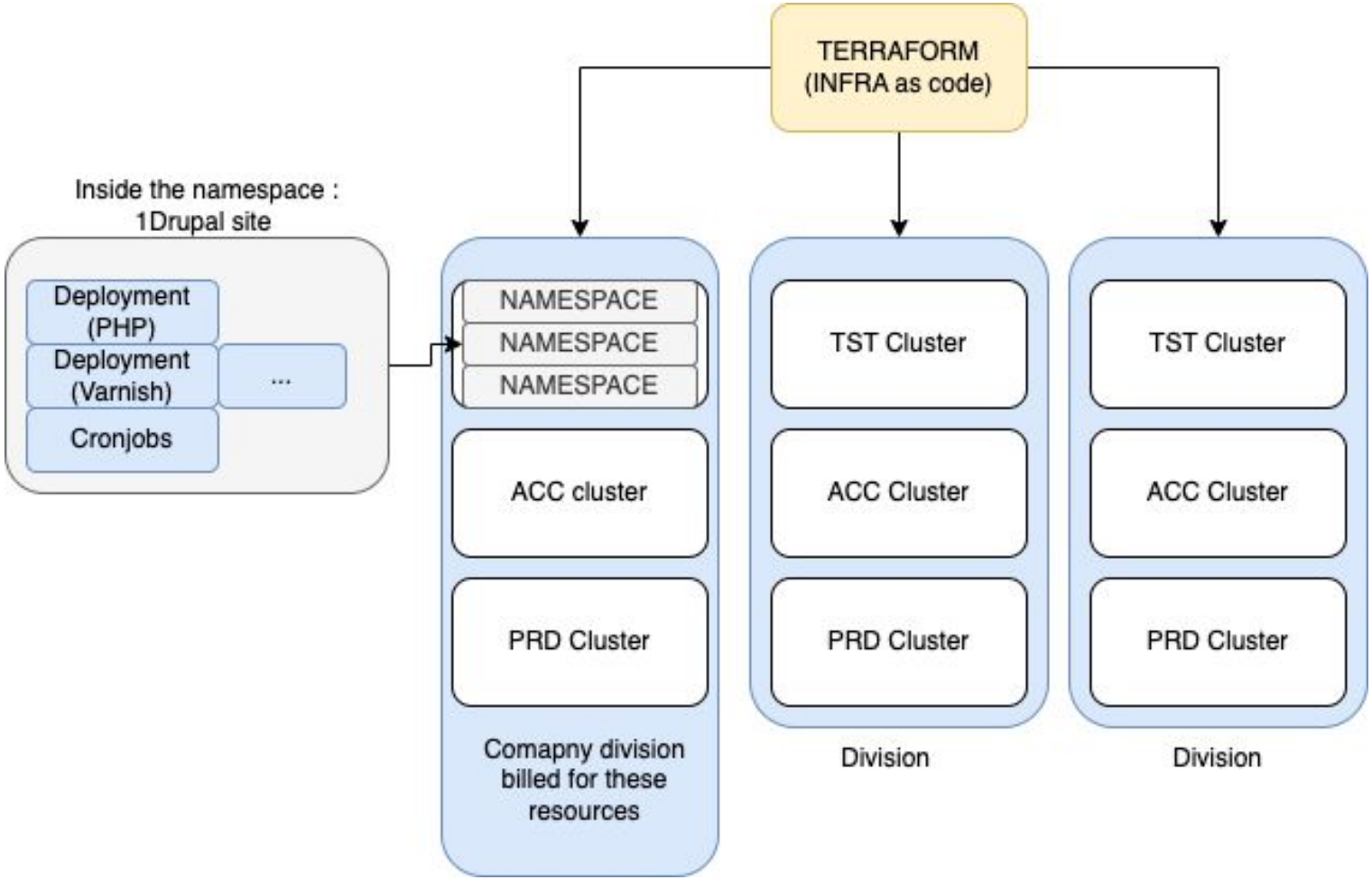
Kubernetes CLI To Manage Your Clusters In Style!



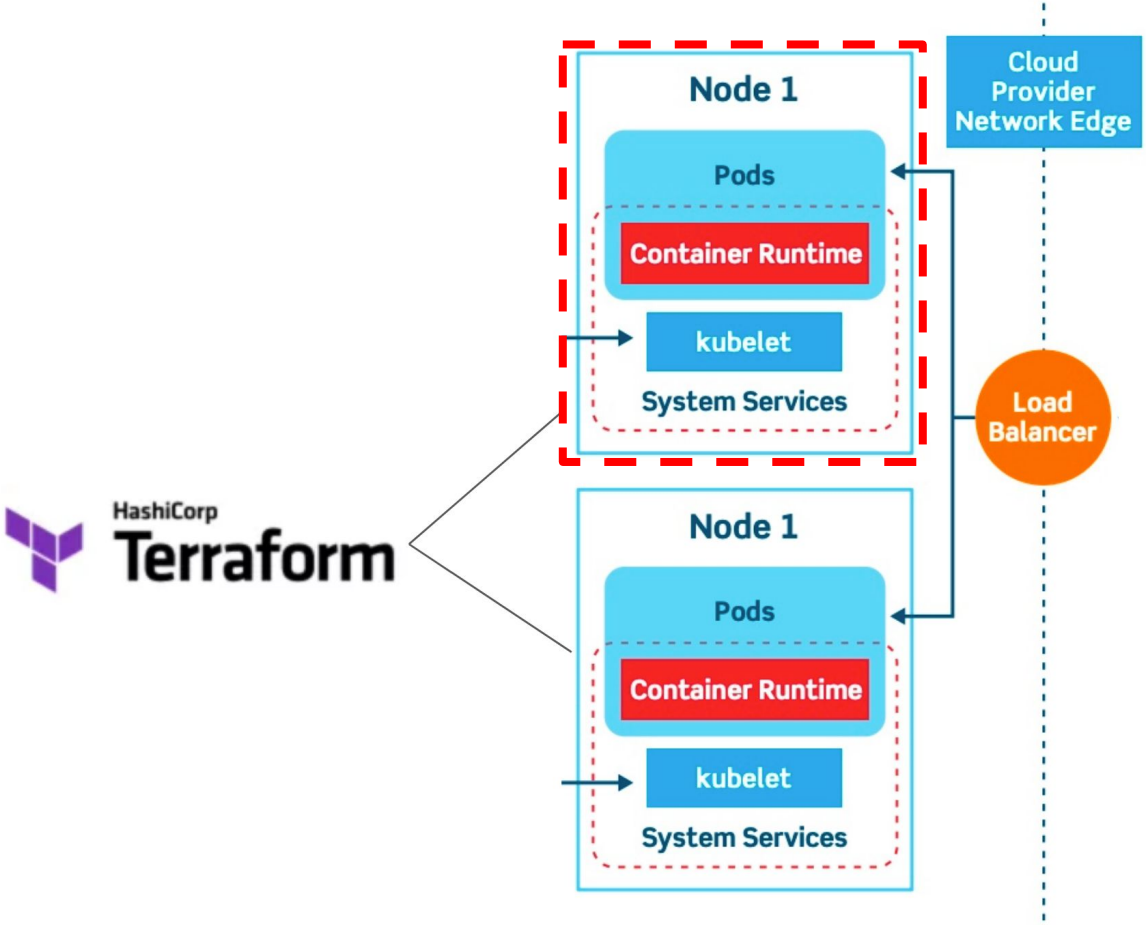


RANCHER
BY SUSE

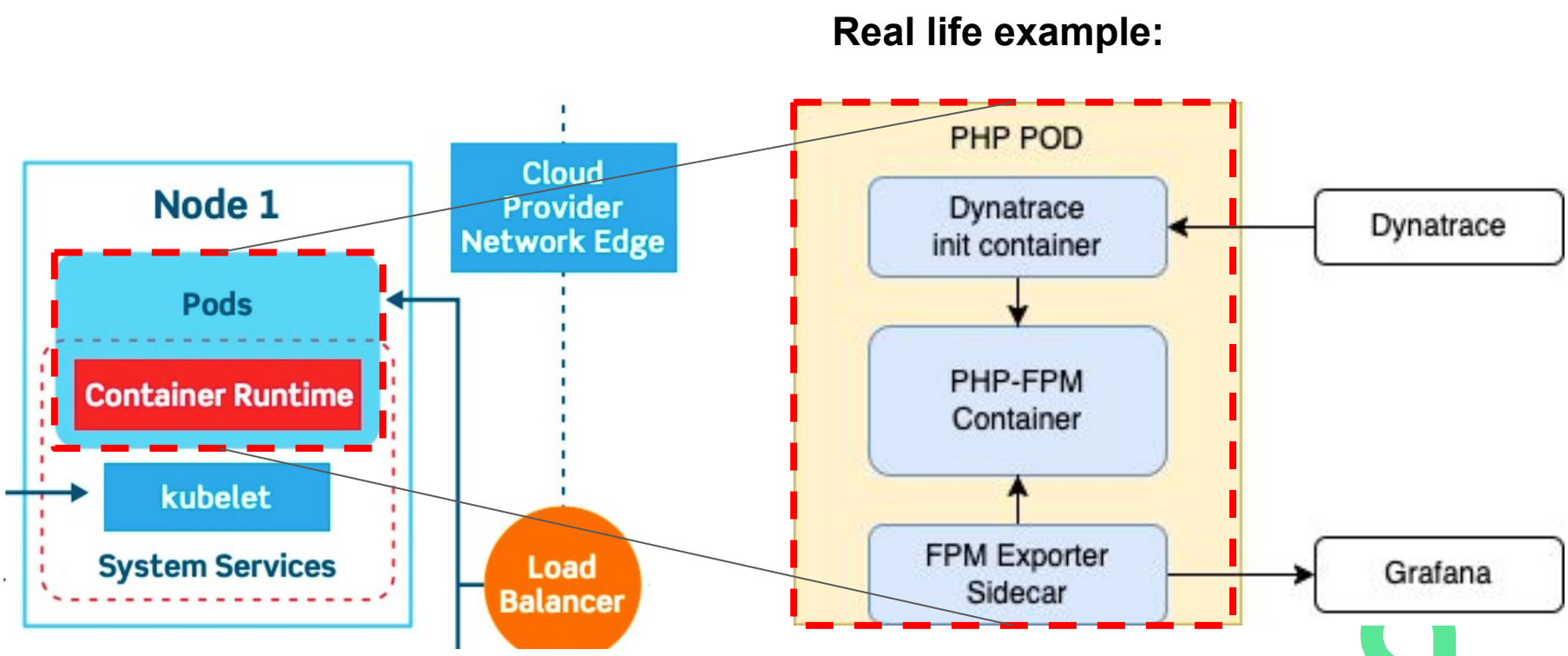
Namespaces


TERRAFORM
(INFRA as code)
Inside the namespace :
1Drupal site
Deployment
(PHP)
Deployment
(Varnish)
...
Cronjobs
NAMESPACE
NAMESPACE
NAMESPACE
ACC cluster
PRD Cluster
Comapny division
billed for these
resources
TST Cluster
ACC Cluster
PRD Cluster
Division
TST Cluster
ACC Cluster
PRD Cluster
Division

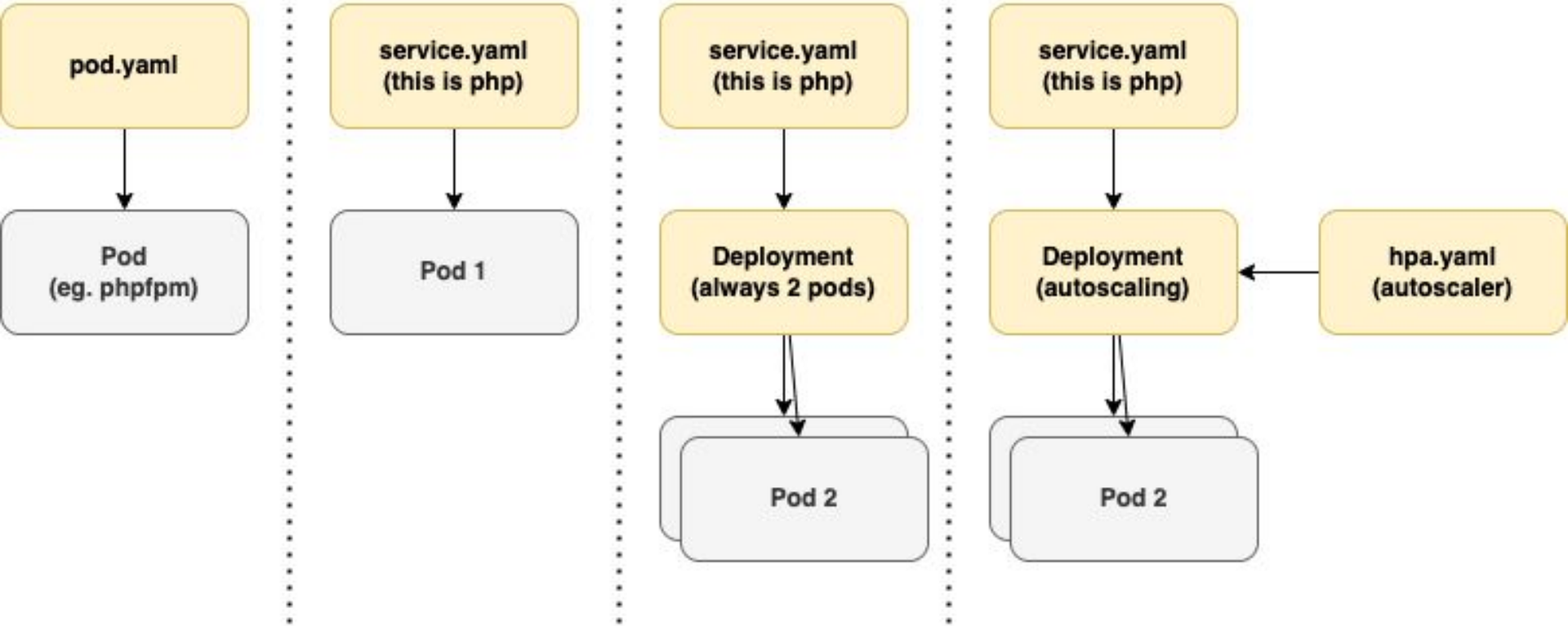
Everything runs on Nodes



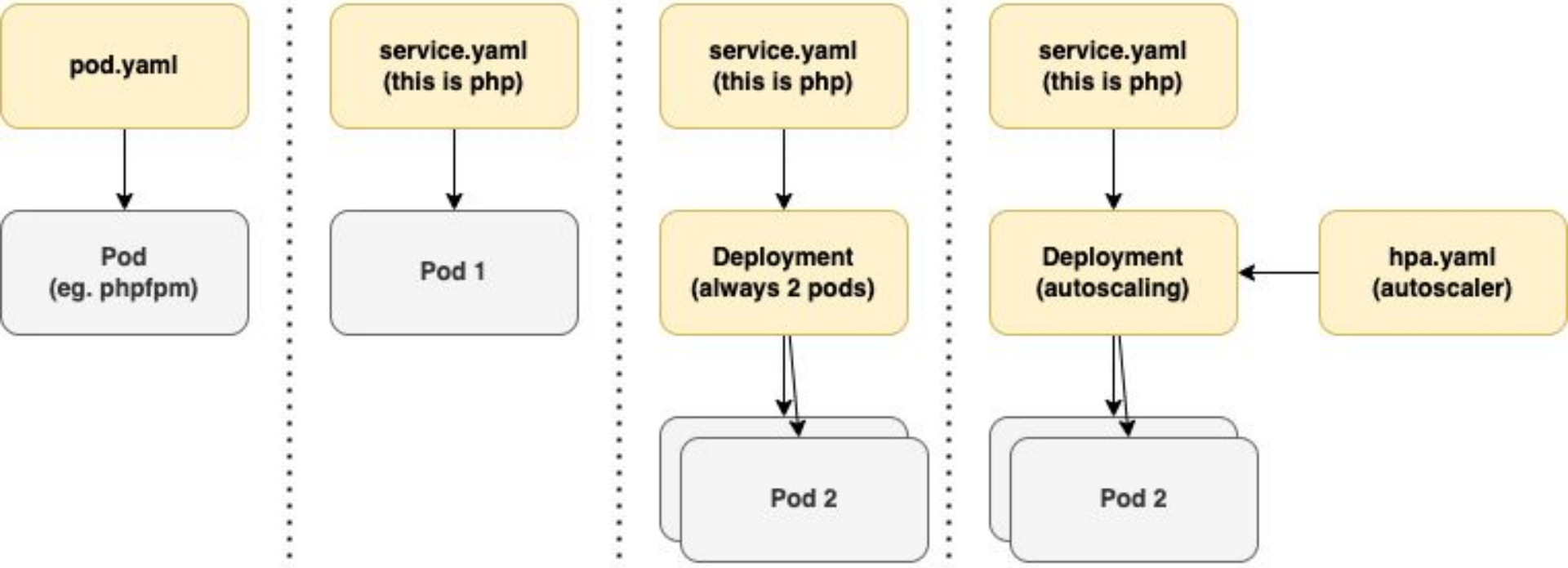
Pod


Real life example:
Node 1
Pods
Container Runtime
kubelet
System Services
Cloud Provider Network Edge
Load Balancer
PHP POD
Dynatrace init container
PHP-FPM Container
FPM Exporter Sidecar
Dynatrace
Grafana

Pod - Service - Deployment - Autoscaling

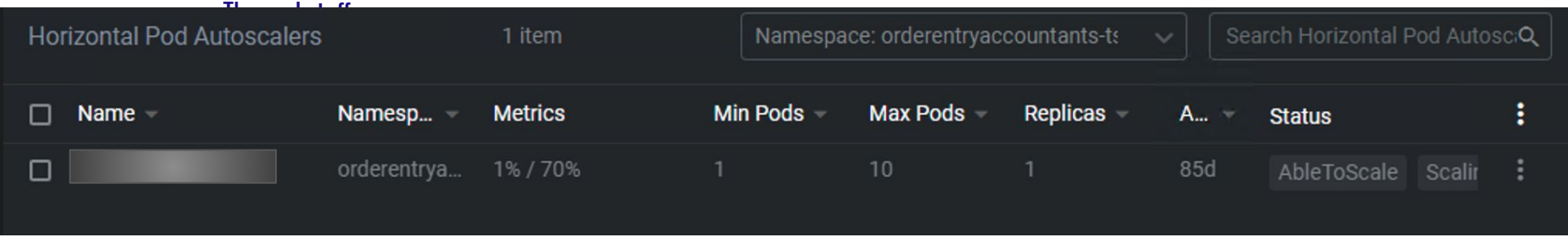


Pod - Service - Deployment - Autoscaling



NEED TO KNOW: HPA needs resource limitations set on pods & CPU < 1 will throttle your system.

Kubernetes - autoscaling



Defined the pods as a deployment
With resources

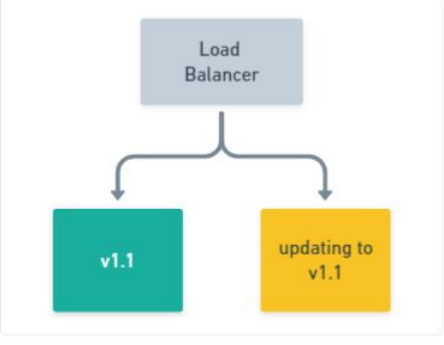
```yaml
resources:
  requests:
    cpu: 100m
    memory: "256Mi"
  limits:
    cpu: 1
    memory: "4112Mi"
```

Describe the HPA

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: {{.Release.Name}}-php-fpm-hpa
  namespace: {{.Release.Namespace}}
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: {{.Release.Name}}-php-fpm
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 70
```

Kubernetes - Deployment strategy

✨ more cool stuff 💫

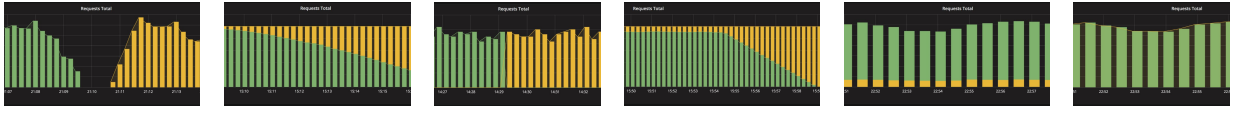


- recreate: terminate the old version and release the new one
- ramped: release a new version on a rolling update fashion, one after the other
- blue/green: release a new version alongside the old version then switch traffic
- canary: release a new version to a subset of users, then proceed to a full rollout
- a/b testing: release a new version to a subset of users in a precise way (HTTP headers, cookie, weight, etc.). This doesn't come out of the box with Kubernetes, it imply extra work to setup a smarter loadbalancing system (Istio, Linkerd, Traeffik, custom nginx/haproxy, etc).
- shadow: release a new version alongside the old version. Incoming traffic is mirrored to the new version and doesn't impact the response.

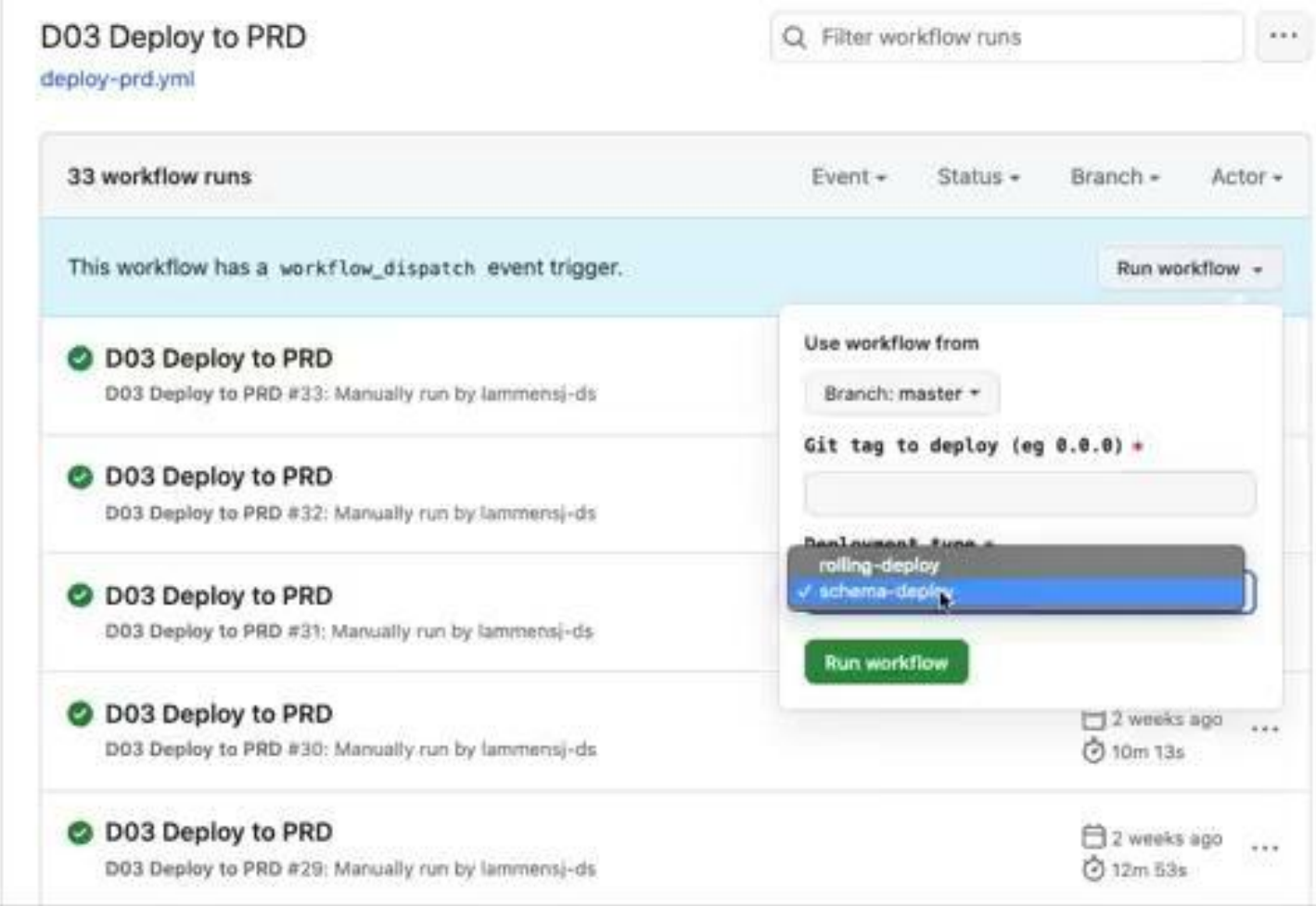
Drupal & Zero downtime deploys

No Schema updates

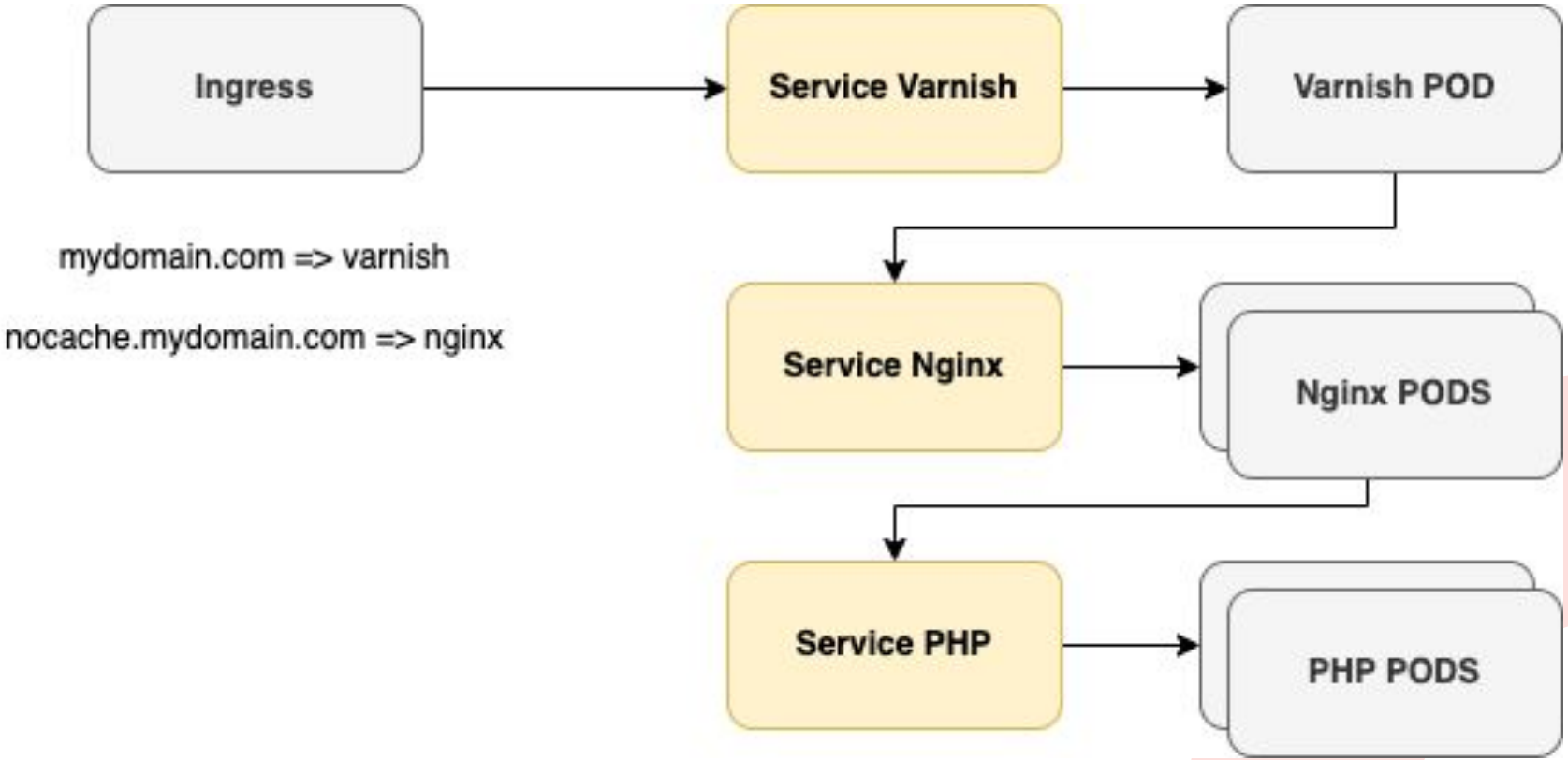
Schema updates



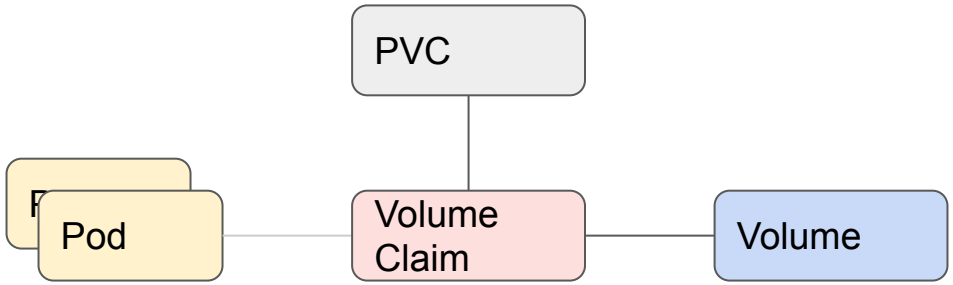
Drupal Zero downtime deploys



Ingress


Ingress
Service Varnish
Varnish POD
mydomain.com => varnish
nocache.mydomain.com => nginx
Service Nginx
Nginx PODS
Service PHP
PHP PODS

Persistent Volume Claim



eg:
/sites/default/files
/tmp
/private

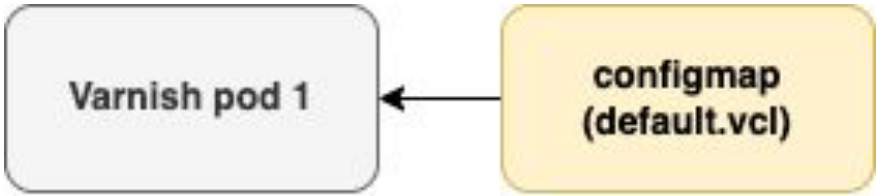
Configmap

MAIN USE

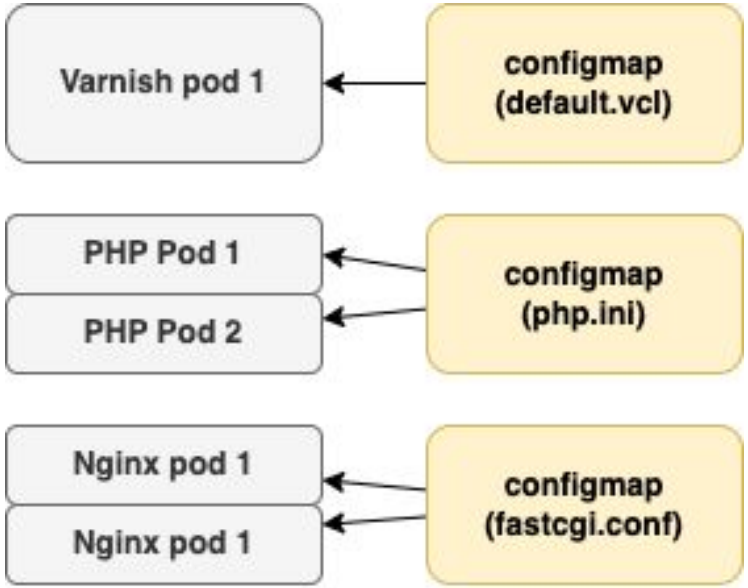
Especially practical if you want to use an

external docker container image (eg memcache or varnish)

with your specific config (without creating your own images)



Configmap

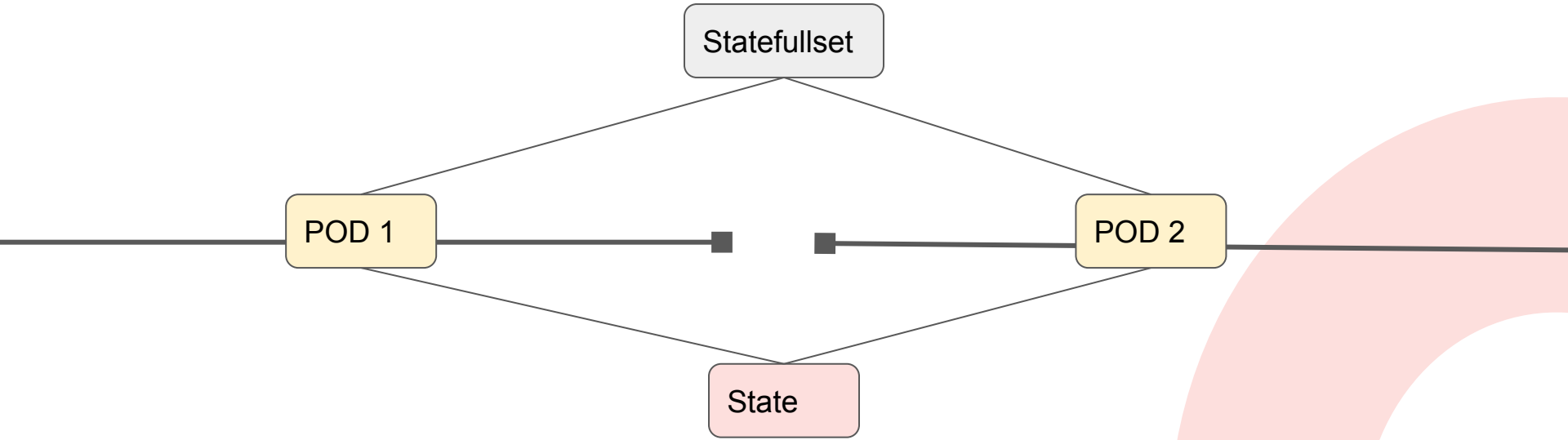


PRACTICAL FOR DEBUGGING

Also super practical to add php.ini or nginx or apache config to your custom images.

This adds the possibility to fiddle on an (obviously not production) environment with the config without a full build + deployment (which is slow).
Change configmap + delete pod (triggers recreate with new config)

StatefullSet

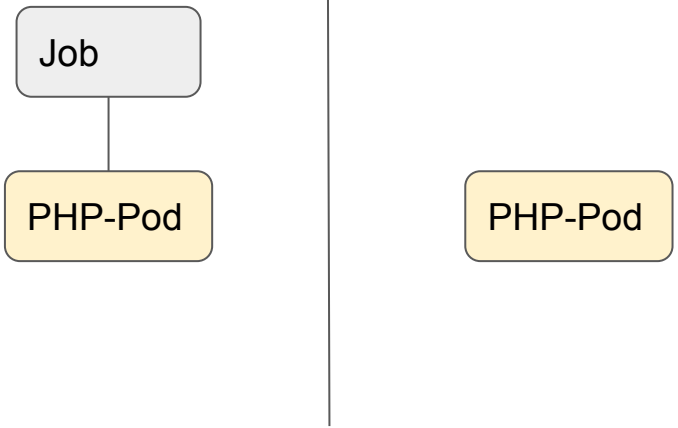


StatefullSet

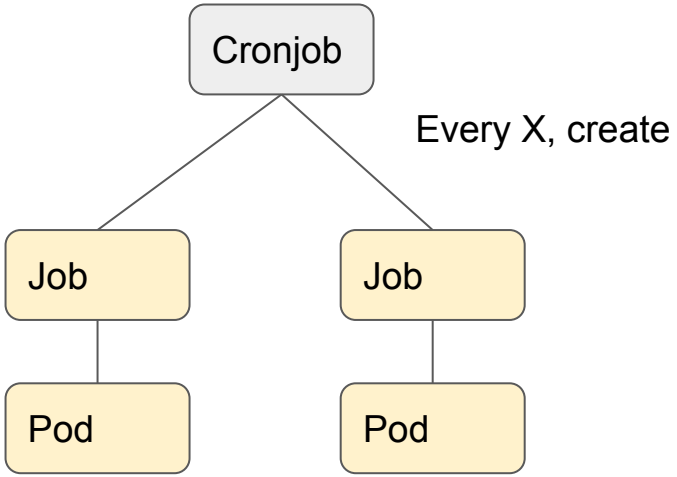
eg: memcache or redis

```yaml
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: {{.Release.Name}}-memcached
  namespace: {{.Release.Namespace}}
spec:
  selector:
    matchLabels:
      app: {{.Release.Name}}-memcached
  serviceName: {{.Release.Name}}-memcached
  replicas: 1
  template:
    metadata:
      labels:
        app: {{.Release.Name}}-memcached
    spec:
      containers:
      - name: {{.Release.Name}}-memcached
        image: memcached
        args: ["-m 64 -p 11211 -p 0.0.0.0"]
        ports:
        - containerPort: [redacted]
          protocol: TCP
        readinessProbe:
          tcpSocket:
            port: [redacted]
          initialDelaySeconds: 5
          periodSeconds: 10
```

Job



Cron



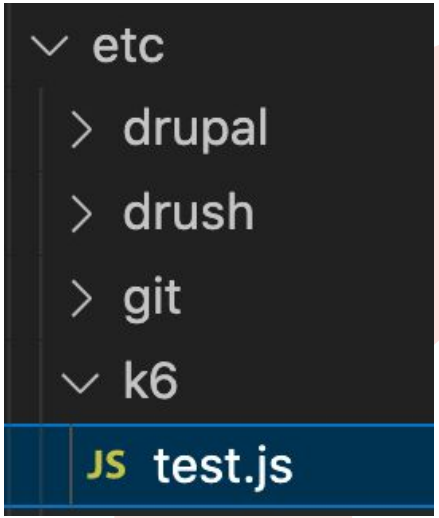
Cron

```yaml
---
apiVersion: batch/v1
kind: CronJob
metadata:
  name: "{{.Release.Name}}-cron"
  namespace: {{.Release.Namespace}}
spec:
  suspend: true
  schedule: "*/5 * * * *"
  concurrencyPolicy: Forbid
  jobTemplate:
    spec:
      template:
        spec:
```

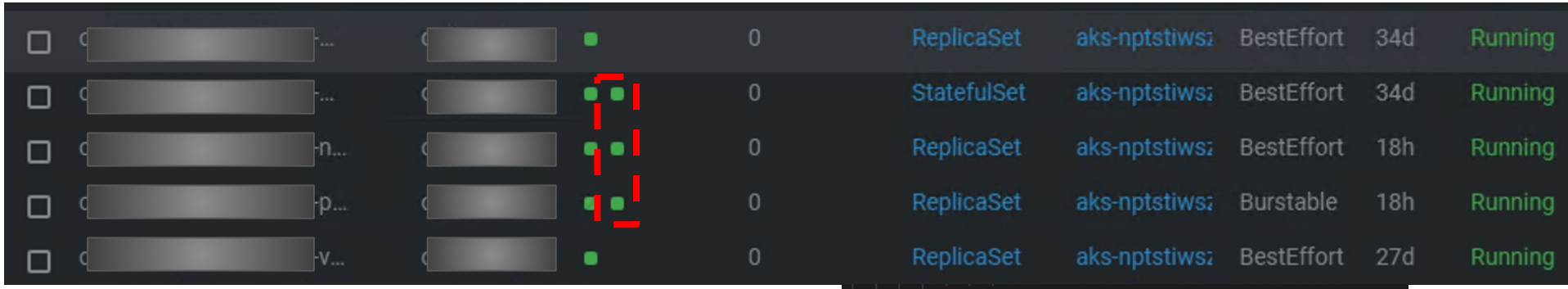
```yaml
    spec:
      template:
        spec:
          {{- with .Values.imagePullSecrets }}
          imagePullSecrets:
            {{- toYaml . | nindent 12 }}
          {{- end }}
          volumes:
          - name: files-vol
            persistentVolumeClaim:
              claimName: {{.Release.Name}}
          - name: private-files-vol
            persistentVolumeClaim:
              claimName: "{{.Release.Name}}-private"
          - name: tmp-files-vol
            persistentVolumeClaim:
              claimName: "{{.Release.Name}}-tmp"
          containers:
          - name: drupal-cron
            image: {{.Values.php_image_name }}:{{ .Values.image.ve
            imagePullPolicy: IfNotPresent
            volumeMounts:
            - mountPath: /var/www/drupal/docroot/sites/default/fil
```

Loadtesting (job)

```yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: "{{.Release.Name}}-loadtest"
  namespace: {{.Release.Namespace}}
spec:
  suspend: true
  schedule: "1 1 * * 1"
  concurrencyPolicy: Forbid
  jobTemplate:
    spec:
      template:
        spec:
          volumes:
          - name: k6-loadtest
            configMap:
              name: k6-loadtest
              items:
                - key: test.js
                  path: test.js
          containers:
          - name: {{.Release.Name}}-k6
            image: loadimpact/k6:latest
            imagePullPolicy: Always
            env:
              - name: HOSTNAME
                value: "{{ .Values.ingress_hostname }}"
            command: ["k6"]
            args: ["run", "/test.js"]
```



Monitoring




```
- name: php-fpm-exporter
  image: hipages/php-fpm_exporter
  ports:
  - containerPort: [redacted]
    protocol: TCP
    name: php-fpm-metrics
  resources:
    requests:
      cpu: 100m
      memory: "128Mi"
    limits:
      cpu: 200m
      memory: "512Mi"
```

new relic. dynatrace Grafana

⚠️ ⚠️ ⚠️ Sidecars Prevent killing of pods.

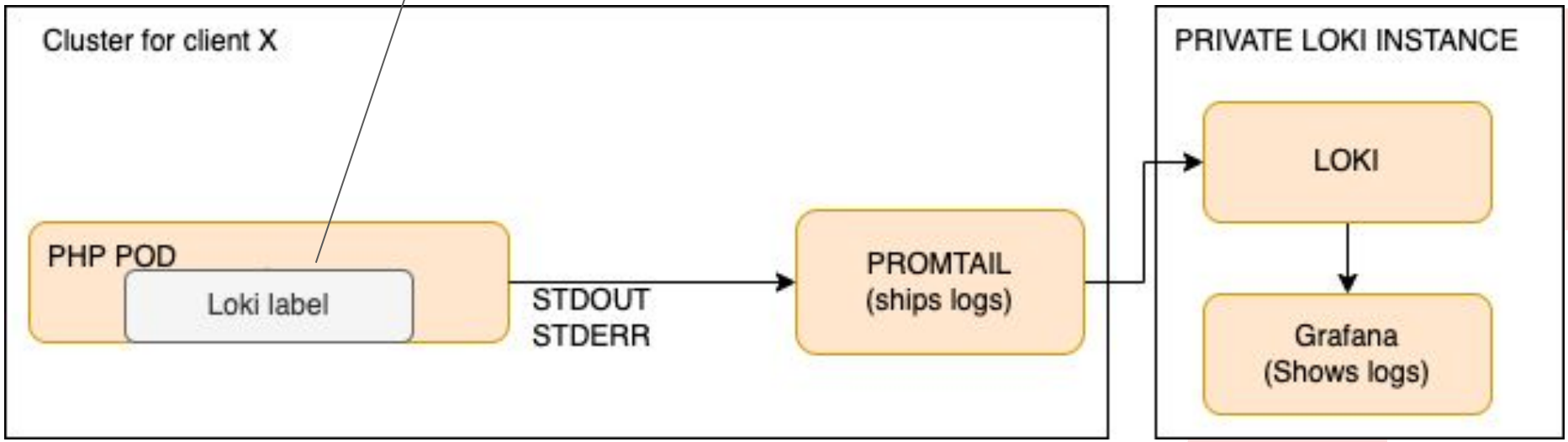
https://medium.com/finnovate-io/how-to-prevent-kubernetes-cron-jobs-with-sidecar-containers-from-getting-stuck-912c0f1497a3

Logs

Grafana

Labels

```yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: "{{.Release.Name}}-cron"
  namespace: {{.Release.Namespace}}
  labels:
    type: "allcronjobs"
    loki: {{.Values.division }}-{{.Values.environment }}
```



Infra as code

Infra as code

drupal-helm-charts-k8s

This is the representation of the above kubernetes concept specifically for (normal) drupal

Pitfalls

Warning

```
containers:
- name: {{.Release.Name}}-varnish
  image: varnish:6.4
```

Fix your version
All of a sudden you're running a newer version of varnish in your cluster , this is something you want to test!

Pitfalls

Warning

```
# Validate helm to not leave a broken state.

- name: DRY RUN
  run: |
    cd helm
    helm upgrade --dry-run --timeout 15m --install -n ${{
inputs.app_key }}-${{ inputs.environment }} -f values-${{
inputs.environment }}.yaml --set deploy.runid=${{ github.run_id }}
--set image.tag=${{ inputs.version }}  --set image.version=${{
inputs.version }}  ${{ inputs.app_key }} .
```

It's not always correct (sometimes deploys still fail) but will fail if your syntax is wrong in your overrides and will prevent you bringing everything down.

Do this before the actual deploy!

Pitfalls


```
      - name: DRY RUN
        run: |
          cd helm
          helm upgrade --dry-run --timeout 15m --install -n ${{
inputs.app_key }}-${{ inputs.environment }} -f values-${{
inputs.environment }}.yaml  --set deploy.runid=${{ github.run_id }}
--set image.tag=${{ inputs.version }}  --set image.version=${{
inputs.version }}  ${{ inputs.app_key }} .
```

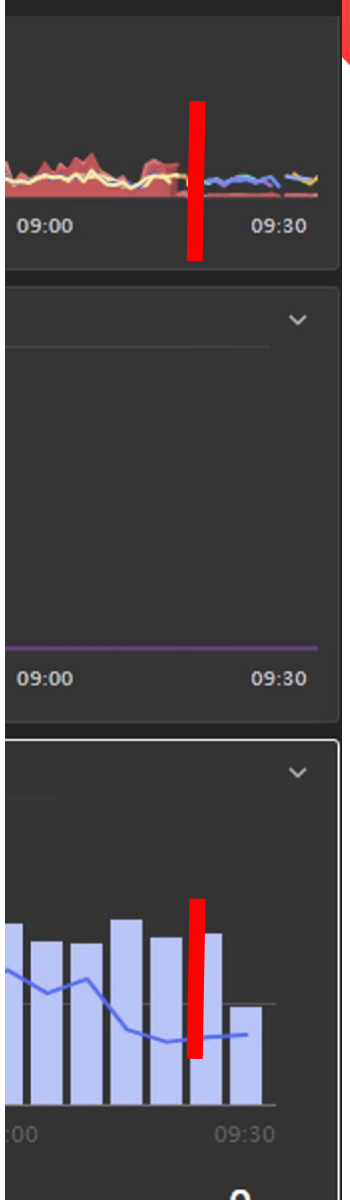
```php
/**
 * Deployment identifier.
 *
 * Drupal's dependency injection container will be automatically invalidated and
 * rebuilt when the Drupal core version changes. When updating contributed or
 * custom code that changes the container, changing this identifier will also
 * allow the container to be invalidated as soon as code is deployed.
 */
define('DEPLOY_UNIQUE_RUN_ID', $_SERVER['DEPLOY_UNIQUE_RUN_ID'] ?? date('Y-m-d-h'));
$settings['deployment_identifier'] = \Drupal::VERSION . '-' . DEPLOY_UNIQUE_RUN_ID;
```

Pitfalls

The first month in PRD
The CPU's were throttling, even with not much load on the systems.

Only after checking Dynatrace and tweaking the CPU from 500m to 1 the throttling disappeared (response time decrease & performance increase).

```
resources:
  requests:
    cpu: 400m
    memory: "256Mi"
  limits:
    cpu: 1
    memory: "4112Mi"
```



Warning

HELM

Pitfalls

regular POD <VS> Deployment

Use a **Deployment** if you want the pods to be there after a cluster rebuild (eg. global outage).

Obviously we learned this the hard way.